# Dokumentimi Teknik

Qëllimi:

Qëllimi është të krijojmë një program i cili na shfaqë tabelen e lojës checkers, lojë e cila luhet me dy lojëtarë dhe në varësi se çfarë inputi jep useri dhe cili user atëherë n’a shfaqet tabela e lojës e ndryshuar dhe ndryshon radha se cili lojëtar ka radhën. Poashtu shikon nëse useri ka shkruar një lëvizje valide, pra në varësi të rregullave të lojës. Qëllimi i lojës është që lojëtari t’i eleminojë gurët e lojëtarit tjetër. Rregullat e lojës janë që lojëtari mund të lëviz gurët gjithmonë diagonalisht dhe mund të zhvendosen në drejtimin përpara (drejt kundërshtarit) në katrorin tjetër. Nëse ka një nga pjesët e kundërshtarit në diagonalen tuaj dhe një hapësirë boshe në anën tjetër, ju mund të hidhni pjesën e kundërshtarit tuaj dhe hiqni pjesën e tij.

Arkitektura:

Klasa Controller: Checkers

Në këtë klasë kemi krijuar disa fusha publike si madhësia, numri i gurëve me ngjyrë të bardhë dhe numri i gurëve me ngjyrë zezë poashtu si një varg dydimensional kemi deklaruar edhe tabelën të cilën do t’a mbushim me gurë me ngjyrë të bardhë apo të zezë dhe një fushë tjetër e cila është e tipit karakter dhe përcakton se radha e kujt është.

Pastaj në konstruktor do të inicializojmë madhësine e tabelës me 8x8 ,numrin e gurëve me ngjyrë të zezë dhe të bardhë me 12 dhe fillimisht do i caktojmë radhën lojëtarit me gurët e bardhë.Pastaj me anë të unazave for do t’a mbushim tabelën me gurë me ngjyrë të bardhë të zezë ose do të lemë hapësira sipas mënyrës se si është tabela në lojën checkers.

Klasa Model: CheckMove

Në këtë klasë do të krijojmë disa metoda të cilat do t’i përdorim për të krijuar rregullat e lojës checkers. Metoda validMove merr dy parametra integer prej ku dëshiron të lëvizë lojëtari dhe ku dëshiron të lëviz.

Sipas metodës që e kemi përcaktuar këtë numër useri e shkruan si numër dhjetor ku numri i dhjetësheve është numri x në tabelë ndërsa y numri i njëshëve. Kështu kur kjo metodë merr këto inpute do merr nga numri xmovefrom/to shifrën e pare të cilën do e fiton kur e pjesëton me dhjetë ndërsa ymovefrom/to shifrën që do e fiton kur merr modulin e inputeve.

Pastaj do shikoj nëse këto shifra janë në rangun prej 1 deri 8 përndryshe kthen false.

Poashtu shikon nëse jemi duke e lëvizur gurin në një pjesë ku nuk ka gurë tjetër sepse kjo nuk lejohet në lojën checkers , brenda kësaj shikojmë nëse brenda një vlere absolute nëse x nga ku duam të lëvizim e zbresim nga x ku dëshirojmë të lëvizim është i barabartë me 1, brenda këtij if shikojmë nëse është radha e lojëtarit të parë dhe nëse y ku dëshirojmë të lëvizim minus y prej ku do lëvizim është i barabartë me 1 pra lëvizim poshtë vetëm për një atëherë do kthehet true dhe nëse është radha e lojëtarit të dytë i cili është poshtë y ku dëshiron të lëviz minus y ku do të lëviz është -1 pra nëse lojëtari i dytë lëviz për një lartë atëherë kthehet true.

Poashtu me një else if shikojmë nëse x prej ku lëvizim minus x ku dëshirojmë të lëvizim është dy poashtu dhe ndryshimi i y është 2 do të kthejmë true vetëm nëse para kësaj është për lojëtarin e parë lojëtari i dytë në diagonale me ku do të lëvizë dhe anasjelltas (pra do të eleminohet lojëtari i dytë apo anasjelltas).

Te metoda tjeter executeMove poashtu marrim dy inpute sikurse te validMove dhe njejtë i nxerrim xfrom, xto, yfrom dhe yto.Pjesen prej ku e lëvizim gurin do të zbrazet poashtu pjesën ku do ta lëvizim do t’a mbushim me varësisht gurin e bardhë apo zi. Poashtu nëse diferenca në mes të x-it nga ku do të lëvizim dhe x ku do të lëvizim është dy nënkupton që kemi mundur të eleminojmë gurin e lojëtarit tjetër prandaj do të zbrazim pjesën në diagonale nga ku kemi qenë në ku do të lëvizim dhe varësisht se cili radhë është do t’a zbresim numrin e gurëve.

Metoda gameOver() e cila kthen një boolean dhe shikon nëse loja ka mbaruar në varësi të gurëve që kemi, pra nëse numri i gurëve të bardhë apo të zi është i barabartë me zero.

Klasa Output: OutputCheckers

Në klasën output kemi një metode winnerIs() që kthen një string se cili është lojëtari fitues pra nëse numri i gurëve të lojëtarit të dytë është zeri fitues është lojëtari i pare dhe anasjelltas. Poashtu kemi edhe metoden printBoard() e cila printon tabelen e lojes checkers duke kaluar me dy for ne çdo pozitë të tabelës.

Klasa Input : InputCheckers

Kjo klasë ka një metodë getNextMove(0 e cila mund të throw exception konkretisht nëse nuk shkruajmë një input. Fillimisht inicializojmë një Scanner, pastaj në varësi të fushës whosemove nësë është w nënkupton që është radha e lojëtarit të pare­ përndryshe është radha e lojëtarit të dytë. Pastaj do deklarojmë një variabël të tipit boolean false, dhe më pas nëse një kjo variable është false që nënkupton nëse një lëvizje legale nuk është shkruar, atëhere do të hyjmë brenda këtij loop-i dhe marrim një integer movefrom dhe nje integer move to. Dhe pastaj me një if do shikojmë nëse lëvizja është valide atëhere do t’a ekzekutojmë dhe do të ndryshojmë vlerën e boolean true. Përndryshe do të shfaqim që ka qenë një lëvizje invalide dhe do kërkojmë për një input tjetër. Poashtu do të ndryshojmë gradualisht radhën nëse është radha e lojëtarit të pare atëherë do të jetë radha e lojëtarit të dytë përndryshe është radha e lojëtarit të pare.

Klasa me metodën main për t’a ekzekutuar programin: TestCheckers

Në klasën TestCheckers kemi krijuar nga një objekt nga klasa OutputCheckers e cila në mënyrë hiearkike ka trashëguar klasët tjera të lartpërmendura dhe fillimisht kemi thirrur metodën printBoard për t’a printuar tabelën.Më pas kemi krijuar një loop i cili vazhon deri sa loja të mbarojë dhe në këtë loop kemi thirrrur një metodë tjetër e cila është getNextMove e cila e merr lëvizjen e shikon nëse është valide dhe nëse po e ekzekuton atë. Pastaj e printojmë përsëri tabelen me metodën printBoard.

Pasi te mbarojë loja dhe programi del nga kjo unazë atëherë me metodën winnerIs nga klasa OutputCheckers kthejmë se cili lojëtar është fituesi.

Diagrami 1: Arkitektura e programit C:

TestSimonSays

Controller:  
Simon

+ Simon()

+ StartGame()

+ HandleInput()

Input:

ColorPad

+ getColor()

+ flashColor()

Model:

Pattern

+ addColor()

+ clearPattern()

+ checkPattern()

Output:

ScoreBoard

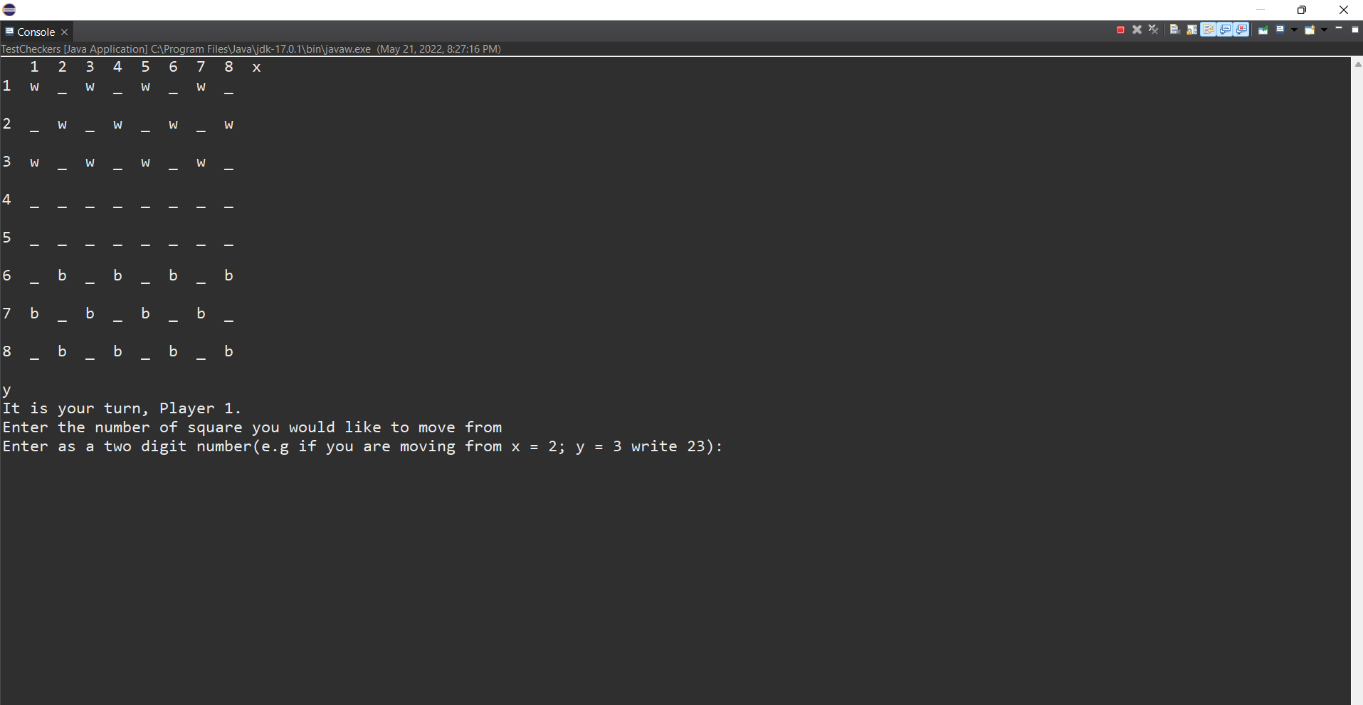
+ increaseScore()

+ resetScore()

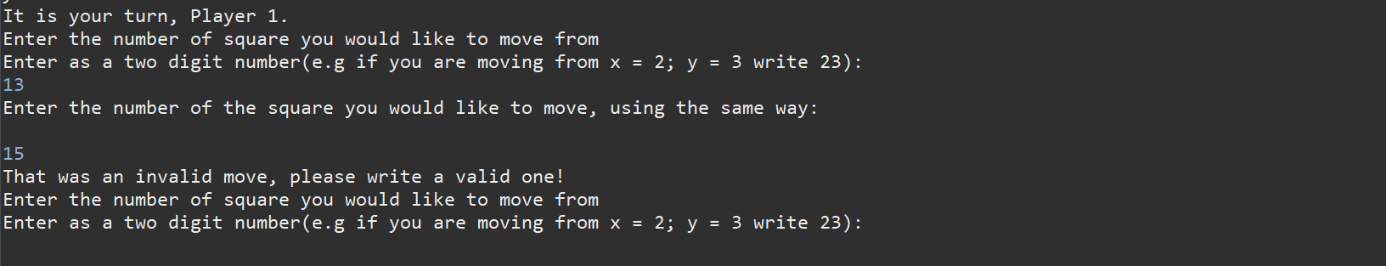
+ getScore

Rezultati:

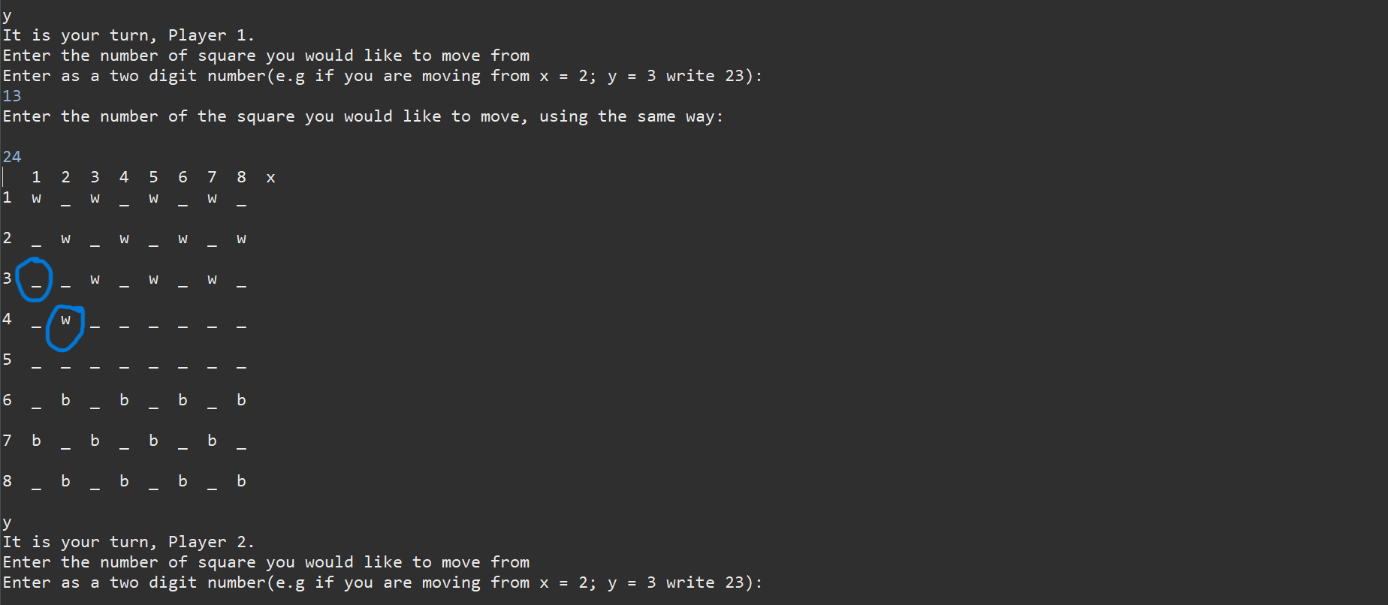
Fillimisht kur e ekzekutojmë klasën TestCheckers në të cilën gjendet metoda main si rezultat do të na shfaqet tabela e lojës checkers dhe radha e lojëtarit dhe si duhet t’a japim ne inputin konkretisht kështu:



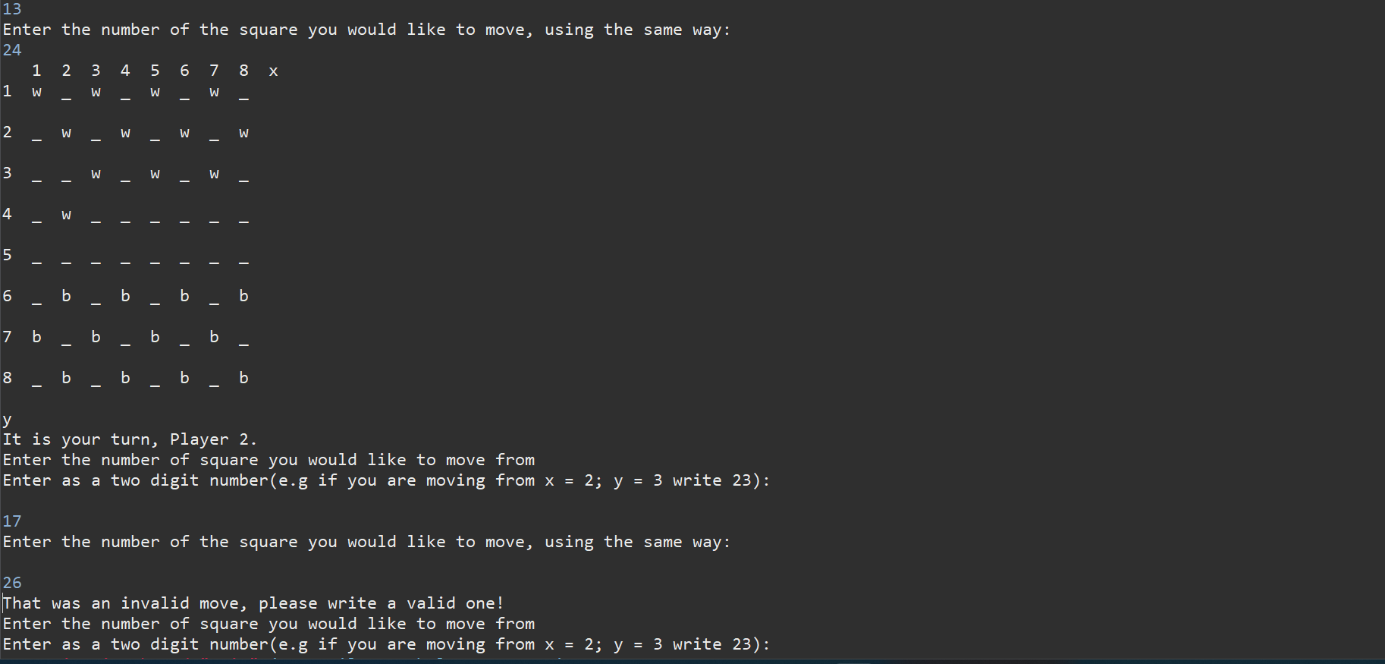
Pastaj do e testojmë se çfarë ndodh nëse shkruajmë një lëvizje që nuk është valide, pra që nuk lejohet me rregullat e lojës së checkers psh: të lëvizim dy poshtë(nëse nuk kemi në diagonale një gurë të lojëtarit tjetër siç e kemi cekur më lart), dy anash, apo diku ku nuk është e lirë :



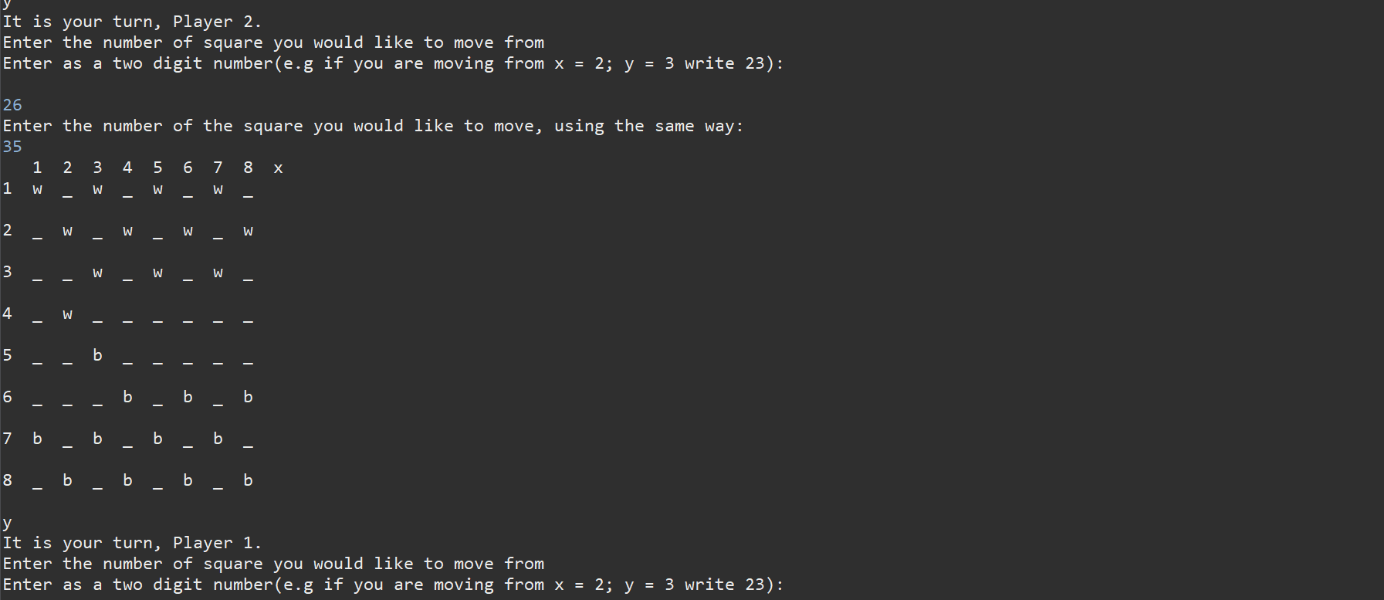
Testojmë se çfarë do ndodh nëse shkruajmë një lëvizje që është valide, pra për një djathtas ose majtas, poshtë ose lartë (në varësi të lojëtarit) dhe ku vendi ku dëshirojmë të lëvizim është i lirë pra nuk është ndonjë gurë tjetër përveç nëse mund t’a eleminojmë atë:



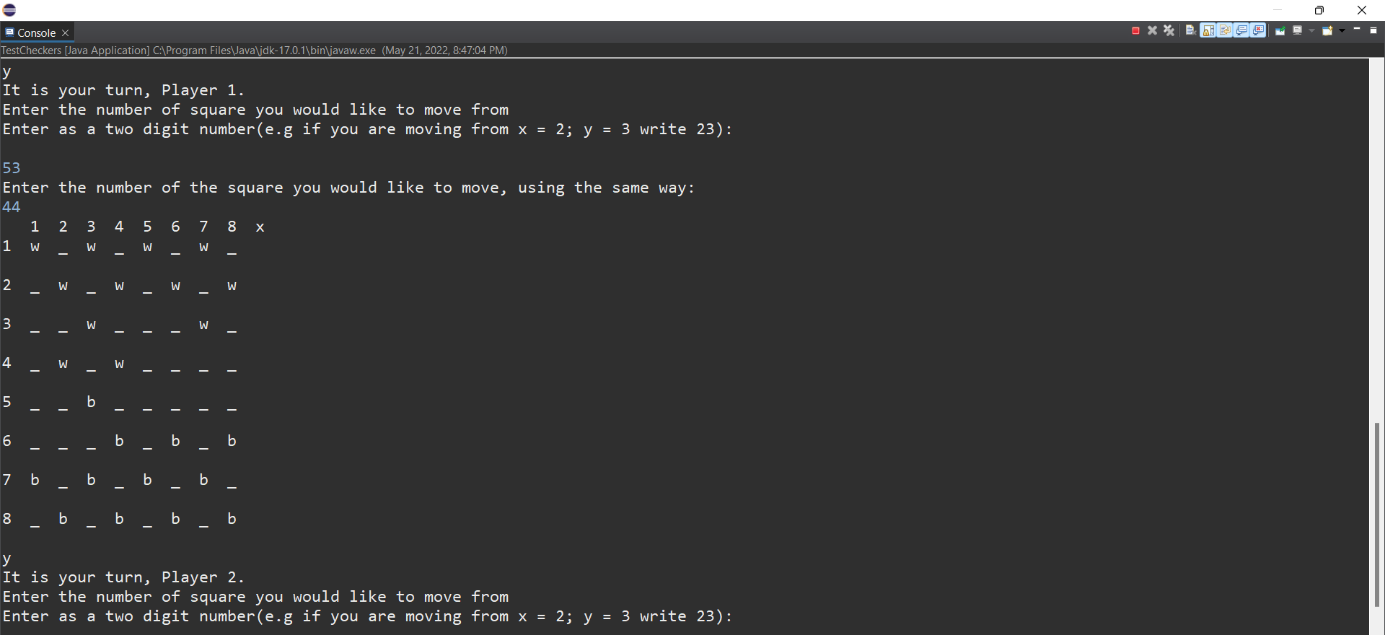
Pra vërejmë që n’a është shfaqur tabela e updatuar me gurin e lëvizur dhe poashtu tashmë është radha e lojëtarit të dytë, provojmë tani rastin kur lëvizim lojëtarin e dytë në një vend ku lëvizim vetëm për një mirëpo aty nuk është e lirë:



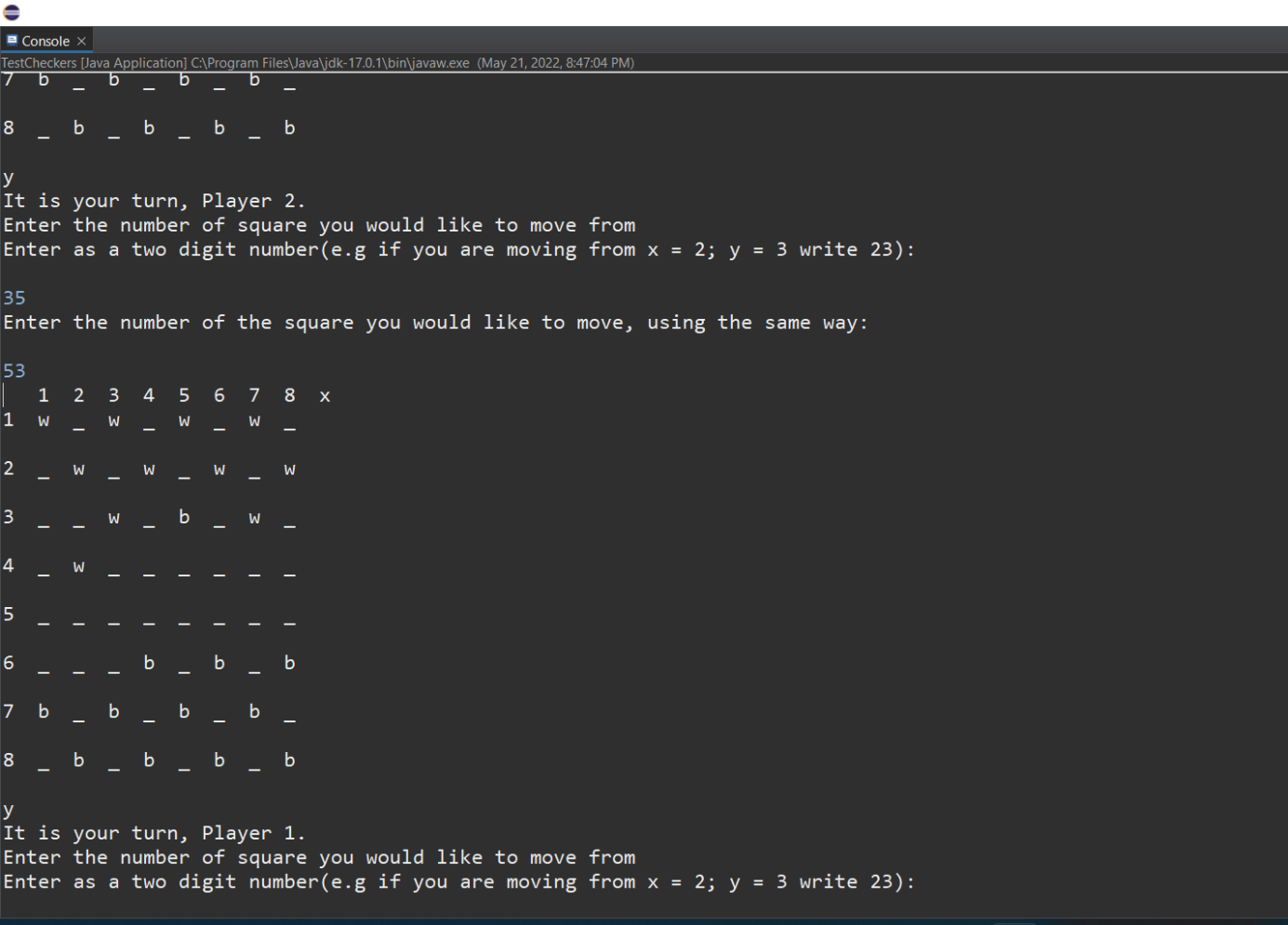
Lëvizim lojëtarin e dytë në një vend ku n’a lejohet sipas rregullave të lojë checkers:



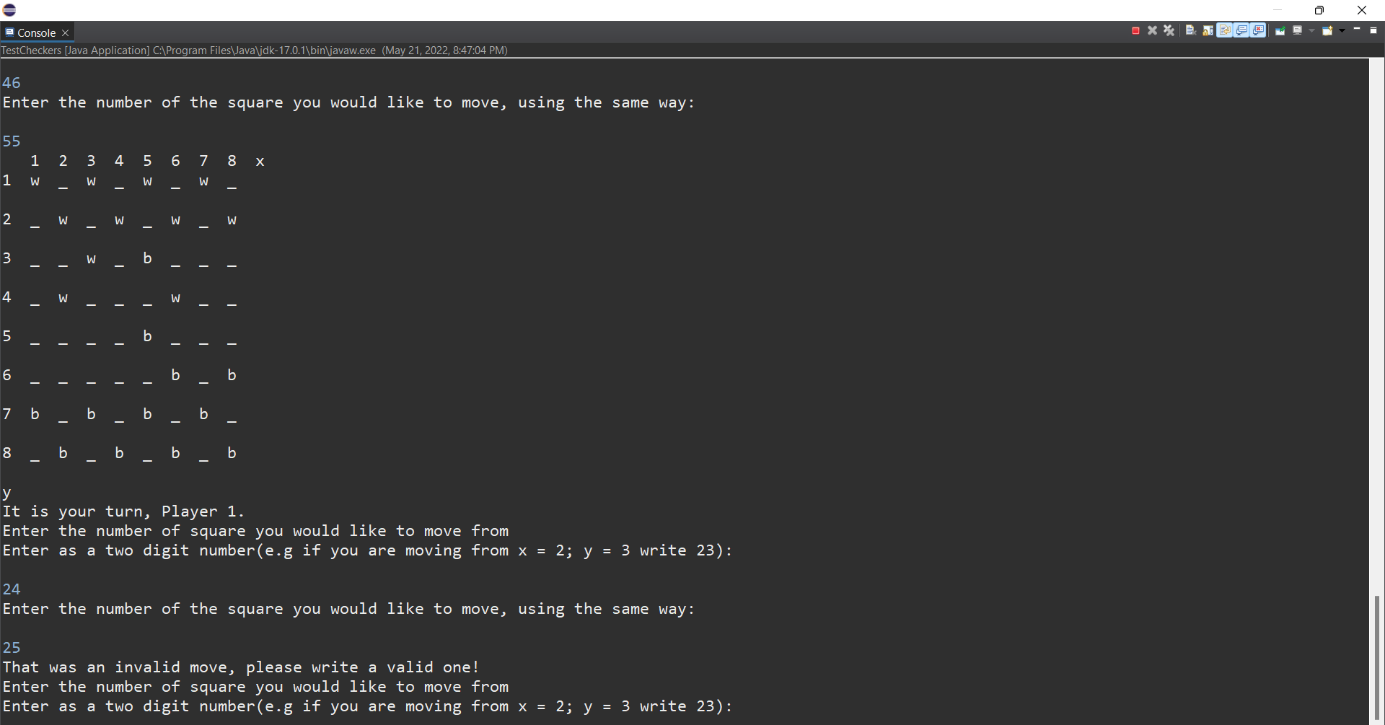
Lëvizim lojëtarin e parë:



Testojmë rastin kur tentojmë të lëvizim për dy dhe në diagonalen që dëshirojmë të lëvizim një para saj është guri i lojëtarit tjetër (pra eleminojmë gurin e lojëtarit tjetër):



Testojmë edhe një rast tjetër kur provojmë të lëvizin gurin për një më lart ose më poshtë:



Dhe në fund testojmë rastin kur loja përfundon dhe ne na shfaqet se kush është fituesi:

